

DATA HANDLING AND POST-RECONSTRUCTION ANALYSIS AT NEXT GENERATION HEP EXPERIMENTS

M. FISCHLER, S. LAMMEL

Fermilab, Batavia, IL 60510, U.S.A.

Abstract: A new generation of experiments in high energy physics is approaching. With the approval of the Large Hadron Collider, LHC, at CERN and the revised Main Injector project at Fermilab, high statistics experiments will start operation within 5 to 10 years. With luminosities of 10^{33} and 10^{34} per cm^2 per second, and several hundred thousand readout channels, data from these experiments most likely cannot be handled and analysed using the traditional HEP approaches. The CAP group at Fermilab is investigating different approaches to data handling and organization for the post-reconstruction phase. We present a discussion of the different approaches considered, their strengths and weaknesses, how they integrate with hierarchical storage, and a study of data exchange and sharing between such systems.

1 Approaches to Post-Reconstruction Analysis

The volume of data involved in High Energy Physics experiments will be increasing by a factor of at least twenty within the next 5–10 years. The prospects of applying current analysis methods, relying on improved computing hardware to cope with the increased flood of data, are marginal at best. Bottlenecks, particularly in bandwidths to disk and tape storage, will have large adverse impacts. Current systems are even marginal for present analysis needs: Physics questions requiring scans of large collider data sets already take weeks to answer; presumably, some investigations which would be more difficult are not being pursued at all.

Approaches to analysis can be characterised by the nature of investigations supported efficiently, selection criteria the user can apply, and the data and results delivered. Finite computing resources necessitate focusing on a few approaches, providing systems (hardware and software) which deal well with large data sets in those contexts. The current analysis approach implicitly chooses a focus driven by realities of yesterday's systems: Easy access to the full data of arbitrary events is sacrificed. The goals of an approach specify what will be convenient and efficient, and might include: cutting on and plotting elements of a pre-determined set of event attributes; applying complicated selection criteria to access the full data of a small set of events; and applying sequences of simple cuts to refine event sets. Some of these goals may be contradictory, in the sense that it is hard to provide efficient implementations of all at once. Clearly, it is better to support several approaches than to tune for only one; to the issue of optimally supporting one analysis methodology, we must add the question of how best to integrate multiple approaches.

Currently, experiments organize raw data in banks handled by FORTRAN memory management packages such as ZEBRA and YBOS.¹ This raw data has little direct value for physics analysis, but is the input to a reconstruction pass, which computes objects of interest, like electrons and jets. At Fermilab, this compute-

intensive step is done on large farms of loosely clustered workstations. This derived information is appended to the raw data; the same data management packages manage the newly created banks. The “full event data” is kept in a collection of sequential files; external catalog systems keep track of contents and locations.

In a second step events are sorted by characteristics, such as existence of energetic electron candidates. The sorting is driven by anticipated or observed access patterns. This same knowledge also leads to dropping of infrequently used information—raw or intermediate data. In this step compression factors of over 100 may be achieved. This makes possible early data-intensive analysis, in which the data remains organized via the same memory management packages. Typically, specific information per event is then collected and saved in n-tuples. This information is rapidly available for histogramming and visualization; but access to information other than the collected attributes can be very time consuming and troublesome. In the current collider experiments at Fermilab, early data-intensive steps have become a significant effort limiting analyses. If analysis for the next generation of experiments is based on this approach, new storage capacity and bandwidth technology will have to improve by a factor of 20 just to match the increased data volume; the present limitations are likely to become much worse.

In contrast, several modern approaches choose some model of what the user will want to do, and attempt to achieve those goals in a more efficient and convenient way. One approach, already implemented and popular, is that of PAW/PIAF.² These assume that the experiment can identify those portions of information which users will want to access—and thus which are worth organizing into n-tuples—and investigations consist of combinations of simple cuts, leading to histograms of selections of the n-tuple data. The creators of PAW/PIAF thus focus on providing a convenient interface for expressing cuts, and on optimizing the accumulation of histograms.

The CAP project at Fermilab assumes that users will want to apply complex selections based on data from among a large set of event attributes, and that all reconstruction summary data or even the entire event data may be needed for the selected events. The small set of chosen events will then be subject to further analysis at user workstations. Selection is made efficient by organizing event data into physics objects (electrons, jets, etc.)—most criteria involve only a few such object types so the entire data need not be scanned.

Another approach integrates data management from early in the analysis cycle, down to detailed study of key events on local workstations. This integrated approach (I-A) assumes that selection cuts are simple and change in incremental fashion, but that the user ultimately needs access to the full data of selected events. Knowledge of how selection criteria tend to evolve is incorporated into the data organization, to optimize the anticipated typical analysis cycle.

The common feature in this phase of analysis is the need for high-performance access to large datasets. The tradeoffs in these approaches are among convenient access to full data, efficiency in scanning and selection, complexity and flexibility of criteria supported, and ease in controlling high-statistics histograms and plots. No single approach is “right”—users analyzing different aspects of the physics will benefit from various capabilities.

2 Hardware Configuration Considerations

Although needs depend on the nature of the approaches supported, on the size of data sets, and on the number of users, systems for data-intensive analysis should meet two common requirements: There should be scalability so that as usage increases one need not change over to an unfamiliar type of configuration; and there must be sufficient connectivity to bring scanned data in from storage, deliver it to the processing CPUs, and provide the results where the user needs them.

Any analysis system will need a large body of data on disk, which today costs about \$300/Gbyte including power and control. A large active portion of the Summary Data for a collider experiment will occupy several hundred Gbytes. Some approaches can process single queries at more than single-disk bandwidth. These can benefit from RAID disks (which are currently expensive and inflexible), data parallelism integrated into the approach, and vendor-provided parallel file systems. An example of the latter is IBM's VESTA/pfs, which might obviate the need to construct parallel access mechanisms suitable for a specific approach; however, early CAP efforts³ have found VESTA to be ill-tuned to the needs of event-oriented data mining.

Since the full data for experiments is measured in Terabytes, exclusive reliance on disk-resident data is infeasible. The least expensive form of storage is tapes, accessed and mounted by "carbon-based robots" (operators). For high repositories, one can measure pure media costs. Video-grade Exabyte tapes cost only 60¢/Gbyte, but the maintenance and human costs associated with drive failures are prohibitive. More realistic costs per Gbyte range from \$1.10 for data-grade Exabytes, through \$3.50 for DLT or (on the horizon) \$5.50 for IBM 3590 cartridges. Although the trend is *not* toward declining prices, the cost is low enough to be unimportant. What is important is the costs of servicing mount requests—personnel, mount latency, drive breakage and mis-mounts.

These costs of human-serviced tape mounts make near-lights-out operation of automated tape libraries (ATLs) attractive. Each robotic library can handle thousands of tapes, and mount more than 6000 tapes a day (a staff of 10 operators is worn down at a quarter of that level). Here the hardware cost is dominated by space for tapes, in a large and reliable robot. With equipment available today, this is almost as costly as disk storage; but new higher-density drives are changing the equation. The CAP system has a 2800-cartridge library which is presently being upgraded to use 10 Gbyte IBM 3590 tapes—the cost (including robot, drives, media, and serving computers) will be \$27/Gbyte. This is certainly suitable as an intermediate way to provide automatic access to large data sets; to varying degrees, each approach to analysis can take advantage of such a repository.

To be useful, an ATL must be controlled by some form of hierarchical storage management (HSM) system. Beyond providing a device interface to control the robot, such a system supports a model of user access to the data. For example, to support the view of data as a collection of files, it allows for organization of vast numbers of files, for automated distribution of files among the tape volumes, and for directory infrastructure. Access models may include explicit "get me this file" commands, stub files which are automatically fetched upon access, and nfs or afs file

system interfaces. An important function, and the reason for the term *hierarchical*, is that of buffering data to and from tape and caching frequently used data onto disks, to avoid excessive tape-mount activity. Good HSM software is not easily available and will be expensive; but the cost is small when amortized over tens of Tbytes of ATL storage.

The CPUs, large disk system, and ATL must be coordinated into an integrated whole, to support whichever approaches are implemented on the system.

3 Analysis Methods and Data Organization

The philosophy of an approach—the assumptions made about access patterns and goals—is implemented by organizing data to allow user queries to proceed efficiently. Tricks employed may include isolating data such that typical questions can be answered without bringing in extraneous data; sorting by key attributes (which size-comparison queries may take advantage of), and multiple data copies with orthogonal organization. For each approach, we will present goals and assumptions, discuss data organization, and comment on consequences.

3.1 PAW/PIAF

The PAW/PIAF approach recognizes that n-tuples are a powerful tool for eliminating unnecessary data access and expressing data selection and extraction needs. A fraction of the event data is put into n-tuple format (and kept on disk)—a good deal of physics thought goes into just which information is “hot”, and how it is to be sorted and compactified for optimal access. PIAF pays particular attention to caching frequently accessed data, and to efficiently forming histograms.

Thus the data organization is that of long vectors of one attributes, for example, a vector of electron energies. A query may need to scan only a few of these, to both select and histogram requested data. If few vectors are needed for a query, performance can be excellent without “striping” to parallelize disk access. For such queries, if data is properly laid out on disk, PIAF can be the most efficient tool available. To improve speed even further, lists are cached into main memory. This approach does not directly deal with tertiary (tape) storage issues.

PAW/PIAF supports a simple “query language” to allow convenient expression of selection criteria, and also supports user-written FORTRAN selection routines. As long as a question requires no data outside the pre-determined n-tuples, it can be expressed conveniently and processed rapidly. On the down side, access to data outside the n-tuples remains awkward. And attributes of variable-length are not well supported. The n-tuples can be laid out on disk such that extending the data or adding new ntuples is straightforward—the implementation of PIAF on the D0 Challenge does this, at a very slight cost in scanning speed. PAW and PIAF have chosen to optimize for an analysis style requiring instant access to a small number of attributes. Other, “othogonal” approaches are complementary and valuable.

3.2 CAP

CAP goals were influenced by Computing division experiences and a series of user meetings⁴, focusing on situations where limitations imposed by n-tuple methods impede analysis. These goals lead to assumptions: some users will want full event data, or at least data which is too large to keep on disk for every event; when this complete data is desired, the selected results will be a tiny fraction of the whole data set, suitable for transfer to a local workstation for closer scrutiny; and selection criteria will range from simple cuts to complex conditions based on multiple physics objects in the event. Data involved in event selection is kept on disk; because the amount of data to be scanned is large, methods of exploiting parallel access to many disks in a scalable system are needed. Rapid assembly of disk-resident data for selected events is desired, in instances where the user does not request full data for selected events. Smooth access to data on tape is also required, for extraction of full data for chosen events, but not for event selection. Both smooth access to data on tape (for extraction of full events) and rapid assembly of disk-resident data for selected events are required. Details of CAP data flow are presented elsewhere at this conference.⁵

Data is organized so as to avoid excessive input when evaluating selection criteria, yet minimize fragmentation of disk-resident data (which would impede re-assembling chosen events). To do this, the hierarchical structure intrinsic to physics data is translated to C++ definitions of “physics objects. For example, one type of physics object is a muon; the experiment’s data model defines a structure containing all the attributes of muons, and an event may contain any number of muon objects. C++ objects allow natural support for sophisticated concepts such as variable-length attributes and pointers “linking” one object to another. A “persistent object manager”⁶ permits access to the physics objects almost as easily as to memory-resident variables, and allows for organization into stores of muons, jets, etc. Most of the event summary data is kept in object form on disk, organized by physics object, and available for use in selection criteria. Since even complex queries typically involve only a few types of particles, only a fraction of the disk-resident data need be scanned for a given selection; and stores are striped across several disks to be read in parallel by multiple I/O servers⁵ to optimize performance. Further speed can be achieved by isolating commonly used “popular attributes” into smaller stores (for example, just the 4-momentum vectors of electrons)—queries involving only these might run as quickly as PIAF selections. Full event data can be kept on tape for users desiring it; CAP addresses objects in the automated tape library by persistent pointer as if it were on disk or in memory. To minimize tape access needs, frequently requested “hot events” will be cached on disk.

CAP provides an Event Query Language which extends the familiar PAW style, automating conditions involving multiple physics objects within the event and links between objects. Although user-defined selection functions are supported, the intent is to make EQL flexible enough to support any query desired. A typical condition might look like

```
(15 < jet.muon#1.E+jet.muon#2.E < 20) && electron.Pt>10
```

Although the primary goal is to provide full, summary, or partial data of selected events in user requested formats (Native ZEBRA/YBOS, n-tuples of spec-

ified attributes, and/or C++ objects), CAP will also provide statistics on acceptance/rejection of the criteria. And (via a tie-in to Histo-Scope⁷), various attributes can be plotted for events which pass each stage of cuts.

The DST data for D0 Run1B has been brought into the CAP system and can be “mined” via queries.

At the next stage of CAP development, other approaches will be supported on the same system, preferably sharing access to the same data.

3.3 Integrated Approach

In the integrated approach (I-A), the analysis system takes over data management at an early stage. Information about an event, whether from the data acquisition system, the reconstruction passes, or generated during previous analysis steps, is collected in a database system. Data organization spans levels of workgroup servers and local analysis platforms—each maintains its own database, retrieving missing information from a primary repository. Data granularity is increased as it propagates from the primary repository to the local platform. This will ensure high information density and efficient use of storage media. Thus the approach unifies not only the various types of data produced, but also the collection of storage and CPU systems involved in analysis. Efficient communication between platforms is essential.

The goals are based on the observation that a given analysis step usually examines a tiny fraction of most events—but for a very few events, more data (and sometimes the full event data) should be available. In either case, data transport handling and organization must be transparent. High efficiency is based on assumptions about the nature of analysis efforts and selection criteria: The activity for an analysis project has a fairly long lifetime, and rarely wanders across many local platforms. And physicists tend to tighten selection criteria and analysis cuts more frequently than loosening them, and yet less frequently to change the attributes on which the cuts are based. In the primary data repository, data is organized by physics object, as per CAP. Disk storage acts as cache, with infrequently used data remaining in an ATL. When a question is posed, relevant data propagates transparently to the user’s analysis platform, which maintains its own database. Workgroup servers are an intermediate step: A server with high bandwidth to a group of stations all doing related analysis can intercept most retrievals before they involve the primary repository. For example, a “Roma-TOP” server might hold all physics objects popular for top-physics investigations and full events on all plausible top candidates. Then an Italian group doing TOP physics would require little data retrieval across the Atlantic. At the primary repository, a physics object is accessed in its entirety; more local servers can extract and transmit just the desired attributes.

The objects stored at lower levels in the system contain only specific attributes. The data needed for the local analysis is stored at the user’s analysis platform. Tightening cuts involves only data in the local database and is very fast. Attributes involved in a cut are retained locally for all events, so although loosening a cut may require fetching missing information, the few specific events it needs fetch

data on are determined locally—so slight relaxations of criteria are also efficient. Tunable caching schemes retain frequently accessed information at the workstation or analysis cluster; extension into tape storage at each level is natural as well. Thus while PAW is concerned with data organization within random access storage, and CAP with organization on disk and tape but within a central system, I-A also addresses hierarchies of partial data sets spanning geographically dispersed systems on a network.

Access patterns in the databases can be monitored and used to create groupings of frequently accessed data—commonly used attributes or frequently accessed events—for improved performance. Both PAW and CAP recognize that a single organization by event is inefficient, and each uses a fixed additional organization: n-tuples in PAW, stores of physics objects in CAP. I-A takes this a step further, dynamically creating additional organizations when justified by access patterns. This approach will be investigated using the CAP testbed hardware as a primary repository.

4 Coexistence and Data Sharing

Ideally, several approaches should be supported so physicists may choose the tool appropriate for each investigation. Can we do better than providing disjoint implementations of each approach? Unifying interfaces is at best a cosmetic improvement, assuming that each approach will define a user interface appropriate for its needs. data in suitable format and organization—if these can be unified, then multiple approaches can share one copy of the large bodies of data on disk or tape. Since methods derive efficiency from well-tailored data organization, sharing data may involve compromises between efficiency and space, and may be impractical in some cases.

Data in old-fashioned HEP format is not directly usable by the approaches discussed above. Neither the I-A nor the CAP approach can make direct use of the n-tuple data needed by PAW/PIAF, but both can easily generate n-tuple output and would thus be ideal to load such a system. As loading PAW data becomes easier, n-tuples can be redefined to match evolving analysis needs.

Both the I-A and CAP approach maintain full datasets in a primary store, with event information in object oriented form, and organized by physics object—they can share this primary dataset (and also the CAP "cardinal data" kept on disk for rapid scanning). But when I-A dynamically creates reorganized collections of frequently accessed data, CAP cannot take advantage of them—CAP has no mechanism to handle rapidly changing attribute organization. If the integrated approach creates a large body of data to take advantage of detected access patterns, the fraction of disk-resident data shared might become small. And the intermediate data managed by I-A on workgroup servers and local workstations is of course not shared with the other approaches, which do not extend organization beyond a central system.

Acknowledgments

The goals of the CAP data-mining system were developed by an analysis server committee including input from the Fermilab directorate (T. Nash), D0 and CDF (S. Feuss, B. Hollebeek and others), and the Computing Division (J. Butler, Paul Lebrun and others). The system has been implemented by the HPPC department, including key contributions by Guest Scientists from CBPF (A. Nigri, R. Pecanha and L. Whately). D0 users, and in particular Seung Ahn, have contributed valuable guidance as suggestions for features to make this approach useful. And we thank Adam Para, who has brought PIAF up on the D0 SGI Challenge system at Fermilab, for discussions about PAW/PIAF.

References

1. *ZEBRA Reference Manual*, CERN Program Library Long Writeup Q100/Q101; *YBOS Programmer's Reference Manual*, Fermilab CDF note 156.
2. *PAW—The Complete Reference* and *PAW++ User's Guide*, CERN Program Library Long Writeup Q121, Applications Software Group, CERN (1993).
3. K. Denisenko, E. Paiva, M. Isely, M. Miranda, *Vesta Experiments at Fermilab* Fermilab/IBM CRADA Report, July, 1994.
4. Analysis Server Executive Committee (Fermilab internal document), *CAP PROJECT—Short-term Goals Document*, May, 1994.
5. M. Isely, K. Fidler, *et. al.* *The Computing for Analysis Project—A High Performance Physics Analysis System*, Presented at CHEP 95, Rio de Janeiro, Brazil, September 1995. (Proceedings in this volume.)
6. M. Fischler, M. Isely, A. Nigri and F. Rinaldo, *POPM: A Distributed Query System For High Performance Analysis of Very Large Persistent Object Stores*, submitted to IEEE Hawaii International Conference on System Sciences (Jan 1996).
7. Mark Edel *et. al.*, *Histo-Scope User's Guide*, Fermilab Computing Division Publication SP0034, 1994.